Regression Assignment

1. **Problem statement:**
    a. The problem statement is to compute the Insurance charges for a customer based on their **Age, BMI, No. of children, SEX, Smoker or Not a Smoker**
2. **Understanding:**

    Independent variables: Age, BMI, No. of children, SEX, Smoker or Not a Smoker

    Dependent: Charges
3. **Total Number of rows:** 1337
4. **Total Number of columns:** 6
5. **Preprocessing:**
    a. **As SEX, SMOKER column has String (Nominal data), need to convert into 0/1, thus applied pandas library -> get_dummies method . Thus the dataset became:**

	age	bmi	children	charges	sex_male	smoker_yes
0	19	27.900	0	16884.92400	False	True
1	18	33.770	1	1725.55230	True	False
2	28	33.000	3	4449.46200	True	False
3	33	22.705	0	21984.47061	True	False
4	32	28.880	0	3866.85520	True	False
...	...	...	...	...	...	...
1333	50	30.970	3	10600.54830	True	False
1334	18	31.920	0	2205.98080	False	False
1335	18	36.850	0	1629.83350	False	False
1336	21	25.800	0	2007.94500	False	False
1337	61	29.070	0	29141.36030	False	True

1338 rows × 6 columns

R2_Score:

As we have multiple inputs and output as numerical data,
I have applied MLR, SVM, Decision Tree, Random Forest algorithms here:

Model Name	R^2_score
MLR	0.7894790349867009
SVM	0.6288792857320359
Decision Tree	0.8716934306026609
Random Forest	0.8736226875503907

Tuning the parameters:

SVM:

Kernel	C	Gamma	R^2_score
Linear	0.1	Scale	-0.080959968427891
Poly	0.1	Scale	-0.08830237655410711
Rbf	0.1	Scale	-0.08907451521042731
Sigmoid	0.1	scale	-0.08826991450485111
Linear	10	Scale	0.462468414233968
Poly	10	Scale	0.038716222760231345
Rbf	10	Scale	-0.03227329390671052
Sigmoid	10	scale	0.03930714378274347
Linear	100	Scale	0.6288792857320359
Poly	100	Scale	0.6179569624059795
Rbf	100	Scale	0.3200317832050831
Sigmoid	100	scale	0.5276103546510407
Linear	0.1	auto	-0.080959968427891
Poly	0.1	auto	-0.080959968427891
Rbf	0.1	auto	-0.08907451521042731
Sigmoid	0.1	auto	-0.08826991450485111
Linear	10	auto	0.462468414233968
Poly	10	auto	0.038716222760231345
Rbf	10	auto	-0.03227329390671052
Sigmoid	10	auto	0.03930714378274347
Linear	100	auto	0.6288792857320359
Poly	100	auto	0.6179569624059795
Rbf	100	auto	0.3200317832050831
Sigmoid	100	auto	0.5276103546510407

DecisionTree:

Criterion	Max_depth	Splitter	R^2_score
squared_error	5	*best*	0.8420667783377725
friedman_mse	5	*best*	0.8247592981719993
absolute_error	5	*best*	0.8549334830978529
poisson	5	*best*	0.8370067462772068
squared_error	50	*best*	0.6923136340310512
friedman_mse	50	*best*	0.6930369835887366
absolute_error	50	*best*	0.6862874546706454
Poisson	50	*best*	0.719693906627027
squared_error	5	random	0.8491041436762746
friedman_mse	5	random	0.8006347068373014
absolute_error	5	random	0.867493203036949
poisson	5	random	0.8716934306026609
squared_error	50	random	0.7367163929864847
friedman_mse	50	random	0.6971743652804676
absolute_error	50	random	0.7005039657893241
Poisson	50	random	0.6910698158676895

RandomForest:

Criterion	n_estimators	Max_features	R^2_score
squared_error	50	sqrt	0.870029408972246
absolute_error	50	sqrt	0.8716413352284358
friedman_mse	50	sqrt	0.868283465836858
poisson	50	sqrt	0.8733655509389101
squared_error	50	Log2	0.8621383503359108
absolute_error	50	Log2	0.8736226875503907
friedman_mse	50	Log2	0.8629236318262976
poisson	50	Log2	0.8715022263954666
squared_error	100	sqrt	0.8682671249870114
absolute_error	100	sqrt	0.873183586394964
friedman_mse	100	sqrt	0.8678844813076738
poisson	100	sqrt	0.8694685934948518
squared_error	100	Log2	0.8694726783046982
absolute_error	100	Log2	0.8708500790901526
friedman_mse	100	Log2	0.868583614709796

poisson	100	Log2	0.870665321846255

Algorithm chosen:

The model built using **Decision Tree** algorithm has the best R^2_score when compared to other algorithms.